PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : H04L 29/06

A1

(11) International Publication Number: **WO 00/18078**

(43) International Publication Date: 30 March 2000 (30.03.00)

(21) International Application Number: PCT/CA99/00838

(22) International Filing Date: 16 September 1999 (16.09.99)

(30) Priority Data:
09/154,699 17 September 1998 (17.09.98) US

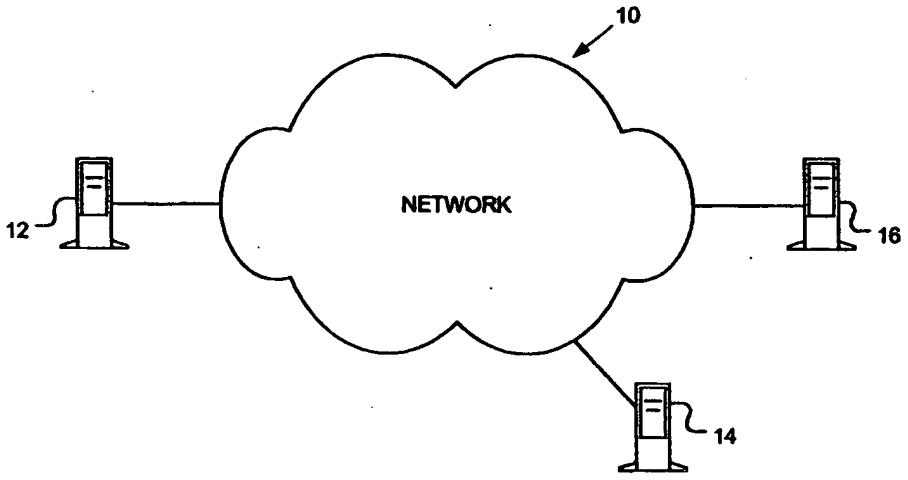
(71)(72) Applicant and Inventor: SOPUCH, David, J. [CA/CA]; 2269 Lakeshore West, 31st Floor, Unit 3, Toronto, Ontario M8V 3X6 (CA).

(74) Agent: ZISCHKA, Matthew; Smart & Biggar, 438 University Avenue, Suite 1500, Box 111, Toronto, Ontario M5G 2K8 (CA).

(81) Designated States: CA, JP, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: SECURE MESSAGE EXCHANGE METHOD USING INTERMEDIARIES



(57) Abstract

A method of providing a message from a first computing device to a second computing device, using an intermediary is disclosed. The first computing device splits the message to be provided into at least two unrelated message portions; enrypts one of the message portions and provides this encrypted to an intermediate computer. The remaining message portion is provided to the second computing device. The second computing device then obtains the first message portion, preferably from the intermediate computer, and combines the message portions to decrypt the message. Preferably, the message is split into the message portion using computationally simple exclusive-OR techniques. As well, preferably the first message portion is encrypted using the widely supported secure socket layer encryption. Using this method, an operator at the intermediate device cannot obtain the message. A third party can only obtain the message by decrypting the encrypted first message portion and obtaining the second message portion. The method may easily be used to split a message into three or more message portions and provided to the second, recipient computer by way of multiple intermediate computers. Devices using the method are also disclosed.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

SECURE MESSAGE EXCHANGE METHOD USING INTERMEDIARIES

FIELD OF THE INVENTION:

The present invention relates to methods and devices for exchanging messages, and more particularly to methods and devices for securely exchanging data between computing devices using at least one intermediary.

BACKGROUND OF THE INVENTION:

In recent years the use of public computer networks to carry sensitive data has become widespread. The best example of such a public computer network is the public Internet. Because of widespread access availability, the Internet is evolving into a preferred communications network. As such, the Internet is being used for the exchange of sensitive data, that may be of a private nature. Recently, the Internet has been heralded as a vehicle facilitating commercial transactions. Because of the sensitivity of financial information, secure communications using the insecure network is a requirement.

As a result, many encryption and decryption methods are being developed. One encryption and decryption mechanisms that has gained popularity is the secure sockets layer ("SSL") method pioneered by Netscape Communications of CA as detailed in Freier, A.O., Karlton, P. and Kocher P. "The SSL Protocol Version 3.0", Netscape Communications, November 18, 1996, and U.S. Patent No. 5,657,390 the contents of both of which are

hereby incorporated by reference. SSL encryption allows an end-user to safely exchange encrypted data using a modified hyper text transfer protocol ("HTTP") session using a temporary session key, which need not be stored or entered by the end-user. Moreover, most currently available Internet-browser applications support SSL encryption. Accordingly, SSL encryption is convenient for end-users.

However, current implementations of SSL encryption require an end-user to communicate with an SSL capable server, such as the Netscape Commerce Server. Many vendors are not able to, or do not wish to administer an SSL capable server. As such, third party intermediaries such as internet service providers have begun operating SSL capable servers for their commercial clients that act as vendors.

End-users may provide sensitive information to the SSL capable servers that vendors may then retrieve, by for example, establishing another SSL session with the SSL capable server. Typically, data received and stored by an SSL server is decrypted and stored at the SSL capable server in plaintext format, until retrieved remotely by the intended message recipient. As such, operators of the SSL capable servers have access to the plaintext message. This may lead to misuse of the sensitive information by these operators.

One known solution addressing this concern requires double encryption of the message by way of another encryption method. For example, the data provided to the SSL server may be encrypted so that decryption is only possible using a key

known to the vendor. This, however, requires the vendor to provide a key to the end-user that must be applied by the end-user using, for example, another software application. This application and the key must be supplied to the end-user prior to SSL session. If the encryption algorithm is complex, the key and software may be quite large and would typically need to be stored at the end-user computing device. All this is quite complex and cumbersome for end-users.

Accordingly, a stream-lined secure method of providing data from a first computing device to a second computing device using an intermediary is desirable.

SUMMARY OF THE INVENTION:

In accordance with an aspect of the present invention, there is provided a method of conveying a message from a first computing device to a second computing device. The method comprises the steps of: a. splitting the message at the first computing device into at least two independent message portions, wherein each message portion is insufficient to form the message and all the message portions are required to form the message; b. encrypting one of the message portions at the first computing device; c. providing the encrypted message portion from the first computing to an intermediate computing device; d. providing the remaining message portions to a second computing device; e. providing the first message portion to the second computing device; and f. re-combining the first message portion and the remaining message portions at the second computing device to form the message.

In accordance with yet another aspect of the present invention, there is provided a computing device comprising: a processor; a computer network interface in communication with the processor; persistent storage memory in communication with the processor, the persistent storage memory comprising processor readable instruction adapting the device to: a. split the message at the first computing device into at least two independent message portions, wherein each message portion is insufficient to form the message and all the message portions are required to form the message; b. encrypt one of the message portions at the computing device; c. provide the encrypted message portion from the computing device to an intermediate computing device using the network interface; and d. provide at least one of the remaining message portions to a second computing device interconnected with the network.

In accordance with yet a further aspect of the invention, there is provided a computer readable medium comprising a software application that, when loaded by a network interconnected computing device adapts the computing device to: a. split a data message at the computing device into at least two message portions, wherein each of the message portions is insufficient to form the message and wherein all the message portions are required to form the message; b. encrypt one of the message portions at the first computing device; c. provide the encrypted message portion from the computing device to an intermediate computing device using the network interface; and d. provide at least one of the remaining message portions to a second computing device

interconnected with the network.

BRIEF DESCRIPTION OF THE DRAWING:

In figures which illustrate, by way of example only, embodiments of the present invention,

FIG. 1 illustrates a plurality of network interconnected computing devices, exemplary of embodiments of the present invention;

FIG. 2 illustrates a preferred architecture of one of the devices of FIG. 1;

FIG. 3 illustrates an exemplary organization of memory at one of the devices of FIG. 1;

FIGS. 4 and 5 are flowcharts of methods exemplary of embodiments of the present invention; and

FIG. 6 illustrates a further arrangement of computing devices, exemplary of an embodiment of the present invention.

DETAILED DESCRIPTION:

FIG. 1 illustrates a plurality of computing devices 12, 14 and 16 exemplary of embodiments of the present invention. Devices 12, 14 and 16 are interconnected by data network 10.

Network 10 is preferably a packet switched data network, such as a network adhering to te internet protocol ("IP"), allowing devices 12, 14 and 16 to exchange data. Data may be exchanged between network interconnected computing devices

using the IP protocol as detailed in RFC 791, by way of intermediate routers (not illustrated). Network 10 may for example, be the public Internet, comprised of numerous smaller physical networks all adhering to the internet protocol. Network 10 could, of course, be any other suitable local area, wide area or other computer network, such as a token ring network, or the like.

Each of devices 12, 14 and 16 is preferably a conventional network client or server computing device such as an intel x86 based computer, or any other suitable computing device. In the illustrated embodiments, computing devices 12, 14, and 16 are architecturally substantially similar.

Device 12 acts as a network based client, that may be permanently or intermittently connected to network 10. The architecture of device 12 is illustrated in FIG. 2. As illustrated, device 12 comprises a processor 18, in communication with persistent storage memory 20, and a network interface 22. Processor 18 may for example, be a conventional intel x86 class processor, a Motorola 68000 series processor, a RISC processor or any other suitable processor known to those skilled in the art. Persistent storage memory 20 preferably comprises a combination of read only memory, random access memory, disk storage, and the like. Additionally, persistent storage memory 20 further preferably comprises a device capable of reading data from a removable storage medium 28, such as a diskette, CD-ROM or the like for storage in other portions of memory 20. Network interface 22 may be an ethernet interface, a modem, an asynchronous transfer mode or

ISDN interface, or any other suitable interface for connecting device 12 to network 10. A monitor 24 and input device 26, such as a keyboard further preferably form part of device 12 allowing input and display of end-user data.

An exemplary organization of persistent storage memory 20 of device 12 is illustrated in FIG. 3. Stored within memory 20 are computer software programs and data that are loaded into working memory of device 12 to permit device 12 to be operable as a network based client computing device. As illustrated, memory 20 stores operating system software 34; application software 36; and data 38. Operating system software 34 may, for example, be Microsoft Windows 95 or 98 software; Microsoft NT Workstation operating system software, UNIX operating system software, or the like. Application software 36 includes network interface software 40, which typically includes an internet protocol suite allowing inter-connection with network 10 and thus communication of operating system 34 with network 10 through the physical network interface 22 (FIG. 1). Application software 36 further preferably includes an internet browser application 42 such as the Microsoft Internet Explorer or Netscape Communicator browser or the like. As such, browser application 42 will be capable of displaying documents written in the hyper-text-markup-language ("HTML"), as for example detailed in C. Musciano, B. Kennedy, HTML: The Definitive Guide, 3ed, (Cambridge, MA: O'Reilly & Associates, 1997), the contents of which are hereby incorporated by reference. Preferably browser application 42 is further capable of executing software applications downloaded through network 10. Most

preferably, browser application 42 is capable of downloading and executing software written in the Javascript or Java programming languages as, for example, more particularly detailed in D. Flannagan, Javascript: The Definitive Guide (Nutshell Handbook) (Cambridge, MA: O'Reilly & Associates, 1997) and P. Niemeyer and J, Peck, Exploring Java, 2ed, (Cambridge, MA: O'Reilly & Associates, 1997), the contents of both of which are hereby incorporated by reference. Such Javascript or Java applications may preferably be downloaded through network 10 into data portion 38 of memory 20 and executed by browser application 42, as required. Additionally, application software 36 may comprise other applications 44 used by an end-user for purposes unrelated to the disclosed methods.

Devices 14 and 16 preferably act as network servers. The organization of memories at devices 14 and 16 and specific architecture of these devices are not illustrated. These are, however, similar to the described architecture of device 12 and organization of memory 20. However, each of devices 14 and 16 need not store nor execute an internet browser application, as device 12 preferably does. Instead, devices 14 and 16 preferably execute and store within their persistent storage memory, network server applications, such as for example an HTTP server application such as the Apache internet server application; the Netscape Commerce Server application, or the Microsoft Back Office software application, or the like. Additionally, the network server application at device 14 further preferably allows the exchange of encrypted messages using one or more known encryption methods. For

example, the server application at device 14 preferably supports encrypted communication between network interconnected devices using the secure sockets layer ("SSL") described above. As will become apparent, device 16 typically need not allow for exchange of encrypted messages. Also stored within persistent memory at devices 14 and 16 are common gateway interface ("CGI") applications or Java applications or other software that may be executed at devices 14 or 16 in response to network contact of these devices. CGI programming techniques are detailed in S. Gundarvan, CGI Programming on the World Wide Web, (Cambridge, MA: O'Reilly & Associates, 1996), the contents of which are hereby incorporated by reference. As will become apparent, also stored within persistent storage memory of device 16 are HTML documents and software in the form of Java applets, applications or Javascript code that may be downloaded and executed by device 12 to facilitate encryption in accordance with methods exemplary of the present invention.

In operation, after causing device 12 to become network interconnected, an end-user at device 12 wishes to securely provide device 16 with a message. For illustration purposes, devices 14 and 16 are assumed to be permanently interconnected with network 10, and identified by at least one uniform resource locator ("URL"). Of course, device 14 and 16 could be connected to network 10, intermittently as required. Device 16 may, for example, be offering acting as an electronic commerce server, accepting and verifying orders for particular products or services. As noted, orders may include sensitive personal and financial information.

The secure provision of the message may better be understood with reference to FIGS. 1, 4 and 5. Steps 400 performed by device 12 are illustrated in FIG.4. Steps 500 performed by device 16 are illustrated in FIG. 5. Specifically, in steps S402 and S502 device 12 contacts server 16 over network 10 using the HTTP protocol and a known URL identifying an HTML page used as a starting point, to establish an HTTP session between devices 12 and 16. Eventually after following one or more HTML links from the initially presented HTML page, the end-user at device 12 will wish to securely provide a message to device 16. Specifically, in step S402 device 12 receives a series of HTML instructions provided by device 16 in step S504 causing device 12 to request information from an end-user to be securely exchanged. For example, device 16 may preferably provide an HTML document including JavaScript code and a Java Applet in step S504 causing device 12 to first present an HTML form for completion by the end-user. The end-user, in turn, completes the form by presenting data such as the end-user's name; address; credit card number; and presses a submit icon or key thus providing the provided Javascript code with the plaintext data acquired, in step S404. For the purposes of this description, the plaintext data acquired through the presentation and completion of the described form will be referred to as M1.

Most preferably, the provided Javascript code or Java Applet now at device 12 further causes device 12 to split the data M1, is a manner exemplary of the present invention once the form has been completed. A portion of the provided Java

Applet and Javascript code is executed once all the data on the input form has been provided and the end-user is ready to submit the data to devices 14 and 16 in steps S406-S410.

The Java Applet executing at device 12 forms two independent data portions C1 and C2 from the submitted, plaintext data, M1, in step S406. C1 and C2 may be considered blocks or streams of ciphertext data. C1 and C2 may be combined to form the plaintext data M1, but individually C1 or C2 do not contain sufficient information to re-create M1. Two such data streams C1 and C2 may for example, be formed by generating a random or pseudo-random bit stream B1 that is bit wise exclusive-OR-ed with the data M1. The pseudo-bit stream may be generated using techniques known to those skilled in the art. One stream is the pseudo-random stream, B1 while the other is the resultant exclusive-OR-ed stream (ie. B1 XOR M1). Advantageously and unlike many conventional known and relatively secure public or private key encryption algorithms, splitting data into two streams is computationally simple. This simplicity allows the required Java Applet or Javascript code to be very small and easily and quickly provided to device 12 from device 16. Other techniques for splitting M1 into two or more separate message streams will be understood by those skilled in the art, and are for example detailed in B. Schneier, Applied Cryptography, Protocols, Algorithms, and Source Code in C, 2ed, (John Wiley & Sons: New York, 1996), or A. Shamir, "How to Share a Secret", Communications of the ACM, Nov. 1979, Vol 22, No. 11, the contents of which are hereby incorporated by reference.

Now, one of the two data streams (C2, for example) is provided in steps S408 and S506 to device 16 over network 10 using, for example, an HTTP connection, typically without encrypting this data stream. This received data stream is stored at device 16, also in step S506.

In step S410 the other of the two streams (C1) is provided to the Javascript code at device 12, which replaces M1 with C1 in the HTML form. Thereafter in step S410, browser application 42, under control of the HTML document provided in step S402, establishes an SSL session with intermediate computing device 14 acting as an SSL capable server, and provides C1 to device 14 using the SSL session. Data provided by way of network 10 during the SSL session is encrypted using an SSL session key, and provided to device 14; and decrypted and stored at device 14, preferably as a file, all using conventional techniques understood by those skilled in the art.

Next, in order to retrieve the plaintext message M1, both ciphertext message streams C1 and C2 are required. Thus, upon receipt of the stream containing C2, device 16 under control of software such as a Java application or Java Applet (not illustrated) may accordingly contact device 14 by, for example, establishing an HTTP or FTP session with device 14 over network 10, and preferably providing a password and identifier; and retrieving the stored file containing C1. While typically, device 16 unlike device 14 is not an SSL capable server, it may include client software capable of retrieving data from device 14 using an SSL session. Thus,

device 16 could establish an SSL session with device 14 to retrieve the file containing C1. Alternatively, device 14 could provide a message containing C1 to device 16 once received. This could be done by device 14 initiating a session and providing the file or by way of electronic mail message, sent to or retrieved by device 16, or in any other suitable manner. Once C1 has been received at device 16, the software application at device 16 may re-assemble M1 from C1 and C2 using the inverse operators used to split M1 into C1 and C2 in step S510. Using the example technique, device 16 may bitwise exclusive-OR C1 with C2 to form M1.

Alternatively, streams C1 and C2 may be retrieved remotely from devices 14 and 16, respectively. For example, an authorized remote user (not illustrated) could establish a connection to network 10, using another computing device and contact device 14, preferably using an SSL session, and device 16 to retrieve C1 and C2.

Additionally, and optionally, in order to discover an error in M1, C1 and C2, M1, C1 and C2 may each be appended with a checksum in the form of a CRC, secure hash algorithm, as detailed in B. Schneier, Applied Cryptography, Protocols, Algorithms, and Source Code in C, 2ed, or the like. Corruption in C1 could thus be detected at device 14 or 16, while corruption of C2 or M1 could be detected at device 16. In response to detecting corruption, device 14 or 16 could request re-transmission of C1, C2 or C1 and C2, from device 12, as required.

As should be appreciated from the above description, in order for a third party to intercept the message M1, the third party will require both C1 and C2. As C1 and C2 are routed to different network interconnected computing devices 14 and 16, typically over different network paths, and often over different physical networks all forming part of network 10, interception of both C1 and C2 on network 10 by a third party is highly unlikely. Further, as C1 is encrypted during transmission, a third party obtaining C1 is further unlikely. Moreover, an operator at server 14 cannot obtain M1, as only C1 has been provided. As there is preferably no statistical correlation between C1 and C2, even a brute force attack on C1 or C2 will not be sufficient to obtain M1. Once M1 has been re-assembled it may be processed as required in step S512 at server 16, or remotely.

As will appreciated, the above example embodiments have been described using a single intermediate computing device. The invention may easily be applied to split the transmitted message into three or more portions, and provide portions to additional intermediaries as illustrated by way of example, with reference to FIG. 6. In the arrangement of FIG. 6, computing device 50 wishes to securely convey a message to computing device 56. Device 50 comprises software similar to that described above, and preferably splits a message M1' into three independent message portions C1', C2' and C3'. This may be done, for example, by splitting message M1' into portions C1' and C2'' using the above described XOR technique. Message C2'' may further be split into message C2' and C3' by again splitting C2'' using the described XOR technique. Message

portion C1' is encrypted and provided by way of a network to device 52. At device 52 it is decrypted and stored. Message portion C2' is optionally also encrypted and provided to device 54, where it is decrypted and stored. Again, SSL sessions between devices 50 and 52 and devices 50 and 54 may facilitate the encrypted exchange of C1' and C2'. Portion C3' is provided by device 50 to device 56, and optionally encrypted. Now, device 56 may obtain portions C1' and C2' from device 52 and 54, respectively. Alternatively, device 54 may obtain message portion C2' from device 52. C1' and C2' could be combined at device 54 and provided to device 56. Alternatively, device 56 could obtain C1' and C2' from device 54 and combine these. In any event, once C1', C2' and C3' are combined at device 56 message M1' may be extracted. Using the example XOR technique, M1'= C1' XOR (C2' XOR C3'). Once again, operators at intermediate devices 52, 54 cannot obtain M1' from message portions C1' and C2'.

As will be appreciated the described method can easily be extended to splitting an initial message M into an arbitrary number of intermediate message portions and using an arbitrary number of intermediate devices.

It will be appreciated that the above described embodiments use the Java or Javascript language and SSL encryption, a person skilled in the art will readily appreciate that the described methods may easily be implemented using other known encryption methods and other computer languages. For example, the described Javascript could be replaced with a compiled C application, executing as

a "plug-in" to the network browser 42 or as part of the browser or other application. Moreover, message portions encrypted using the described SSL sessions could be encrypted using any other symmetric or public key encryption methods. For example, the known Pretty-Good-Privacy application available from Network Associates could be used. As well, while communications with server 16 has been described as not requiring encryption, a person skilled in the art will appreciate that communications with device 16 could also be encrypted.

Similarly, while the organization of software blocks, and data portions have been illustrated as clearly delineated, a person skilled in the art will appreciate that the delineation between blocks and data portions is somewhat arbitrary. Numerous other arrangements of software and data are possible. Similarly, while computing device 12, 14 and 16 have been illustrated as substantially similar, a person skilled in the art will appreciate that, in practice, these are typically quite dissimilar.

It will be further understood that the invention is not limited to the embodiments described herein which are merely illustrative of a preferred embodiments of carrying out the invention, and which are susceptible to modification of form, arrangement of parts, steps, details and order of operation. The invention, rather, is intended to encompass all modifications within its spirit and scope, as defined by the claims.

WHAT IS CLAIMED IS:

1. A method of conveying a message from a first computing device to a second computing device, said method comprising the steps of:

a. splitting said message at said first computing device into at least two independent message portions, wherein each message portion is insufficient to form said message and all said message portions are required to form said message;

b. encrypting one of said message portions at said first computing device;

c. providing said encrypted message portion from said first computing to an intermediate computing device;

d. providing the remaining message portions to a second computing device;

e. providing said first message portion to said second computing device; and

f. re-combining said first message portion and said remaining message portions at said second computing device to form said message.

2. The method of claim 1, wherein said remaining message portions are provided to further intermediate computing

devices prior to step d.

3. The method of claim 2, wherein said first message portion is provided to said second computing device by said intermediate computing device.

4. The method of claim 1, wherein said first, second and intermediate computing devices are interconnected with at least one data network, and wherein said first and remaining message portions are provided to said intermediate and second computing device over different data paths on said network.

5. The method of claim 4, wherein said second computing device and said intermediate computing device are interconnected to different physical networks.

6. The method of claim 1, wherein step e. comprises decrypting said encrypted message portion at said intermediate computing device.

7. The method of claim 1, wherein step a. comprises forming a pseudo-random bit stream at said first computing device, and applying said pseudo-random bit stream to said message to form said second message portion, and wherein said first message portion comprises said pseudo-random bit stream.

8. The method of claim 6, wherein step e. further comprises encrypting said decrypted message portion at said intermediate computing device.

9. The method of claim 1, further comprising the step of

g. obtaining a software application to perform step a. at said first device from said second device.

10. The method of claim 1, wherein said first, second and intermediary computing devices are interconnected with a computer network adhering to an internet protocol, and wherein step c. comprises establishing a connection over said network between said first computing device and said intermediate computing device and said encrypted is provided to said intermediate computing device using said connection.

11. The method of claim 10, wherein data exchanged using said connection is encrypted using a temporary key generated for said connection.

12. The method of claim 11, wherein step e. further comprises establishing a network connection between said first computing device and said second computing device, and wherein said first message portion is provided to said second computing device using said session.

13. The method of claim 1, wherein step e. comprises providing said first message portion to said second computing device as an electronic mail message from said intermediary computing to said second computing device.

14. A computing device comprising:

a processor;

a computer network interface in communication with said processor;

persistent storage memory in communication with said processor, said persistent storage memory comprising processor readable instruction adapting said device to:

a. split said message at said first computing device into at least two independent message portions, wherein each message portion is insufficient to form said message and all said message portions are required to form said message;

b. encrypt one of said message portions at said computing device;

c. provide said encrypted message portion from said computing device to an intermediate computing device using said network interface; and

d. provide at least one of the remaining message portions to a second computing device interconnected with said network.

15. The computing device of claim 14, wherein some of said processor readable instructions are provided to said computing device from said second computing device using said network interface.

16. The computing device of claim 15, wherein said processor readable instructions further comprise a pseudo-random bit stream generator and adapt said processor to apply a pseudo-random bit stream formed by said generator to said data message to form said second message portion, and wherein said first message portion comprises said pseudo-random bit stream.

17. The computing device of claim 16, wherein said network comprises and internet protocol compliant network, and said processor readable instructions further adapt said computing device to communicate over said network using an internet protocol.

18. The device of claim 16, wherein said processor readable instructions further adapt said device to provide said first message portion to said intermediate computer using the http protocol.

19. A computer readable medium comprising a software application that, when loaded by a network interconnected computing device adapts said computing device to:

a. split a data message at said computing device into at least two message portions, wherein each of said message portions is insufficient to form said message and wherein all said message portions are required to form said message;

b. encrypt one of said message portions at said first computing device;

c. provide said encrypted message portion from said computing device to an intermediate computing device using said network interface; and

d. provide at least one of the remaining message portions to a second computing device interconnected with said network.

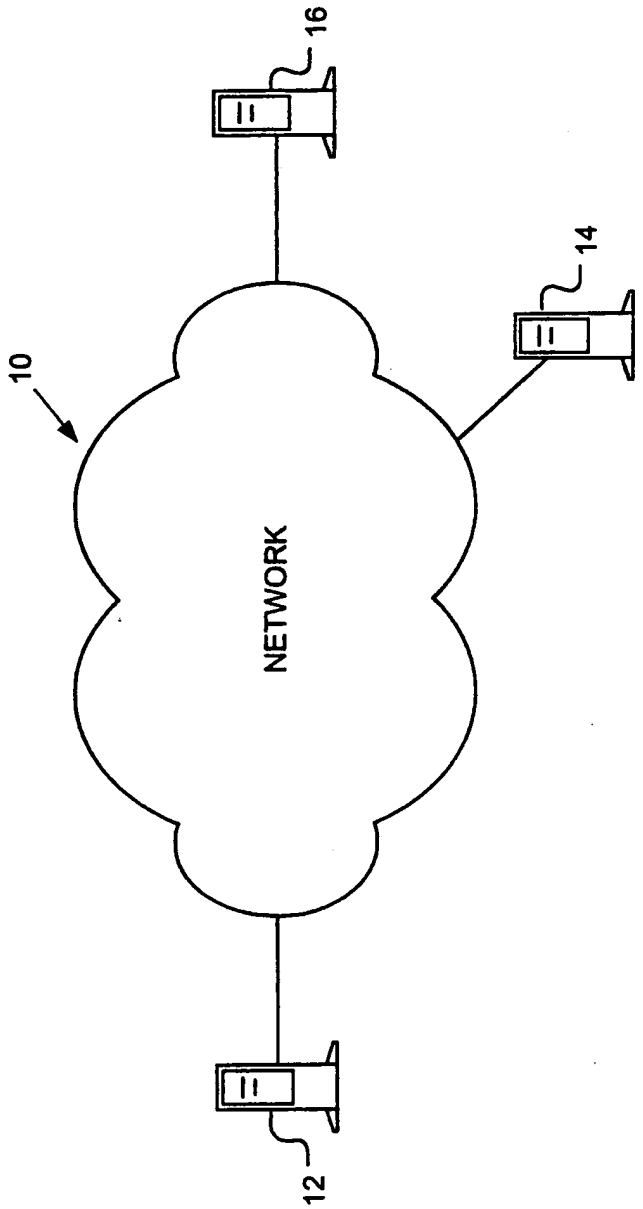


FIG. 1

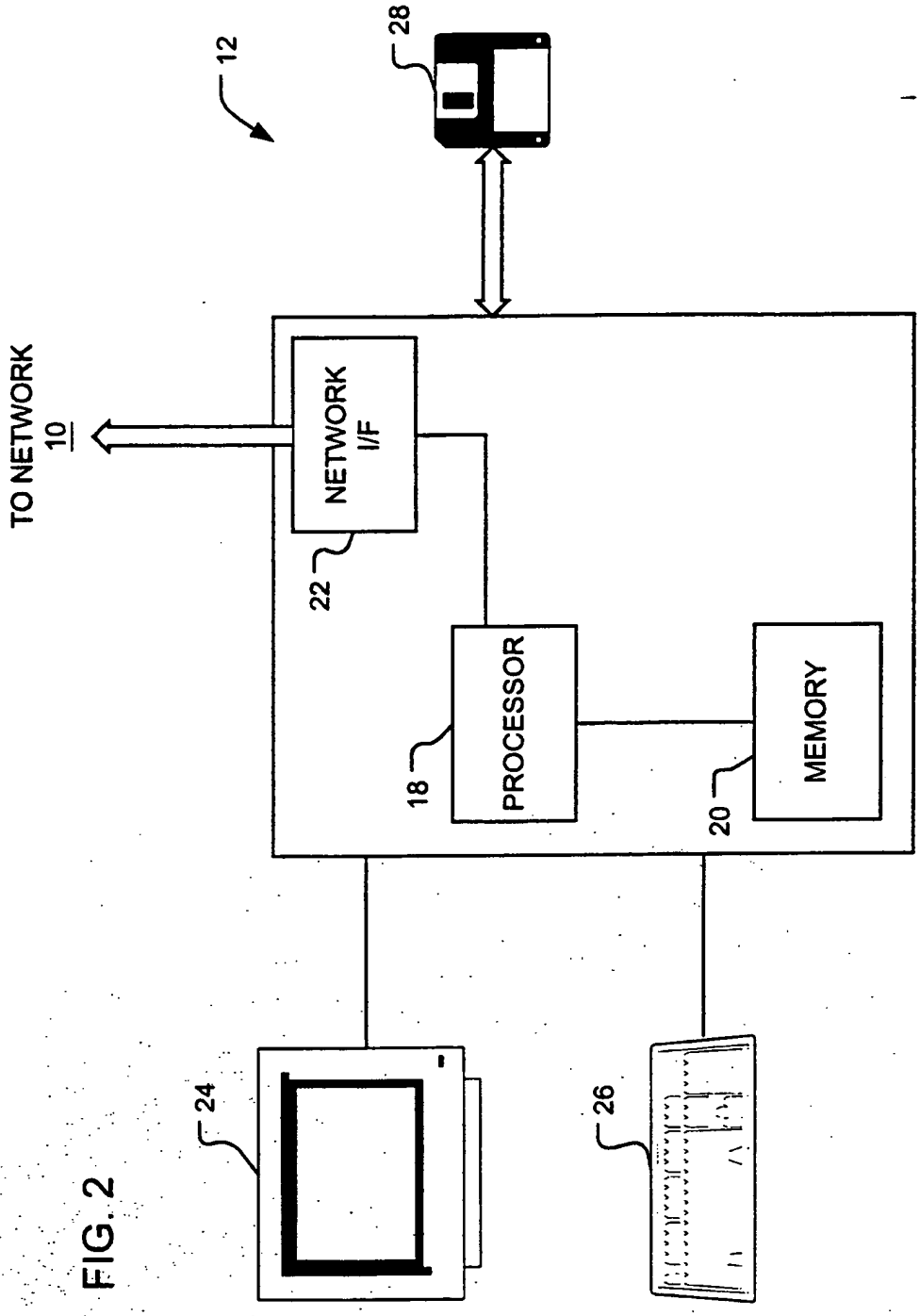


FIG. 2

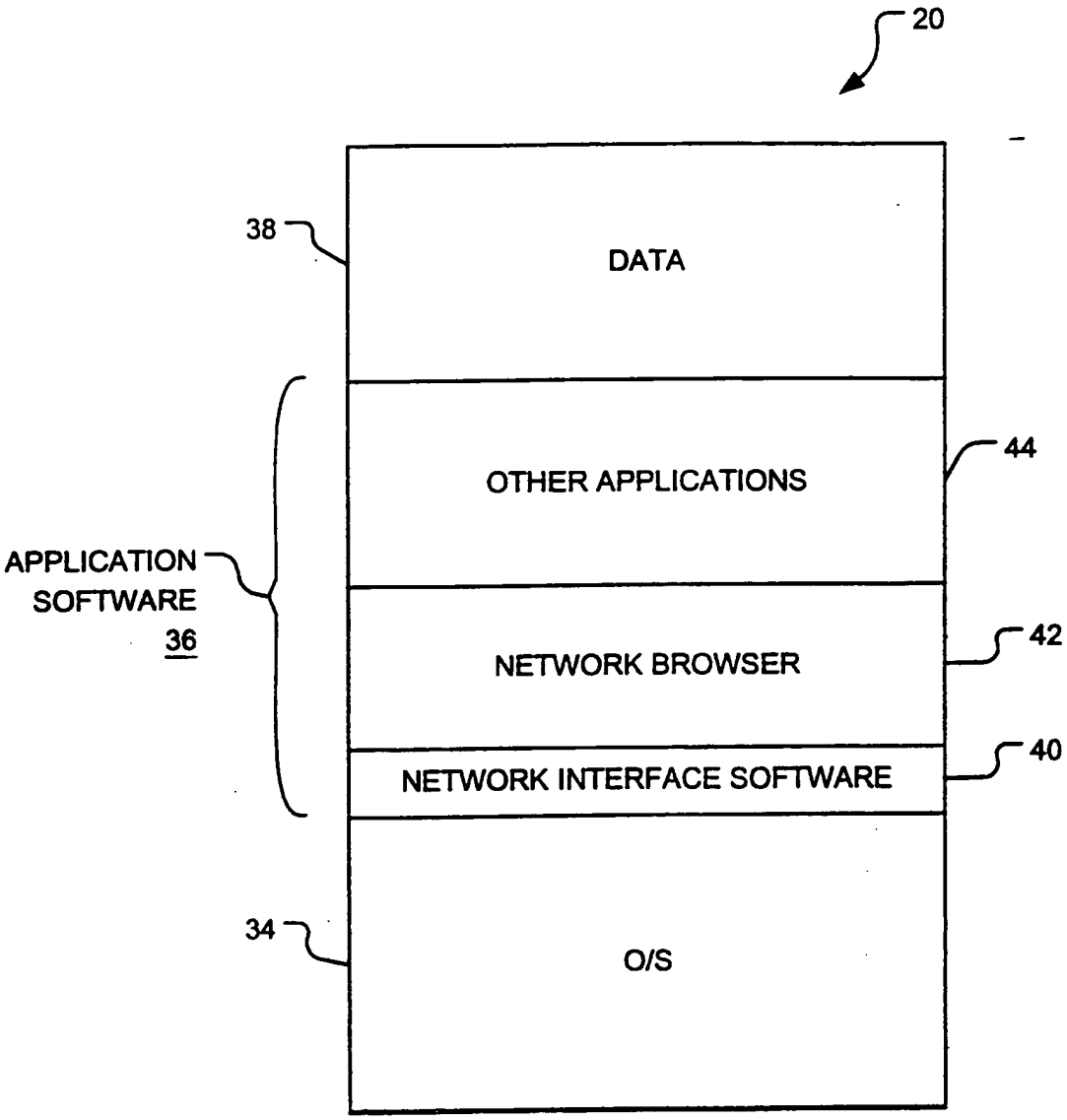

20
38
DATA
OTHER APPLICATIONS
44
APPLICATION SOFTWARE 36
NETWORK BROWSER
42
NETWORK INTERFACE SOFTWARE
40
34
O/S

FIG. 3

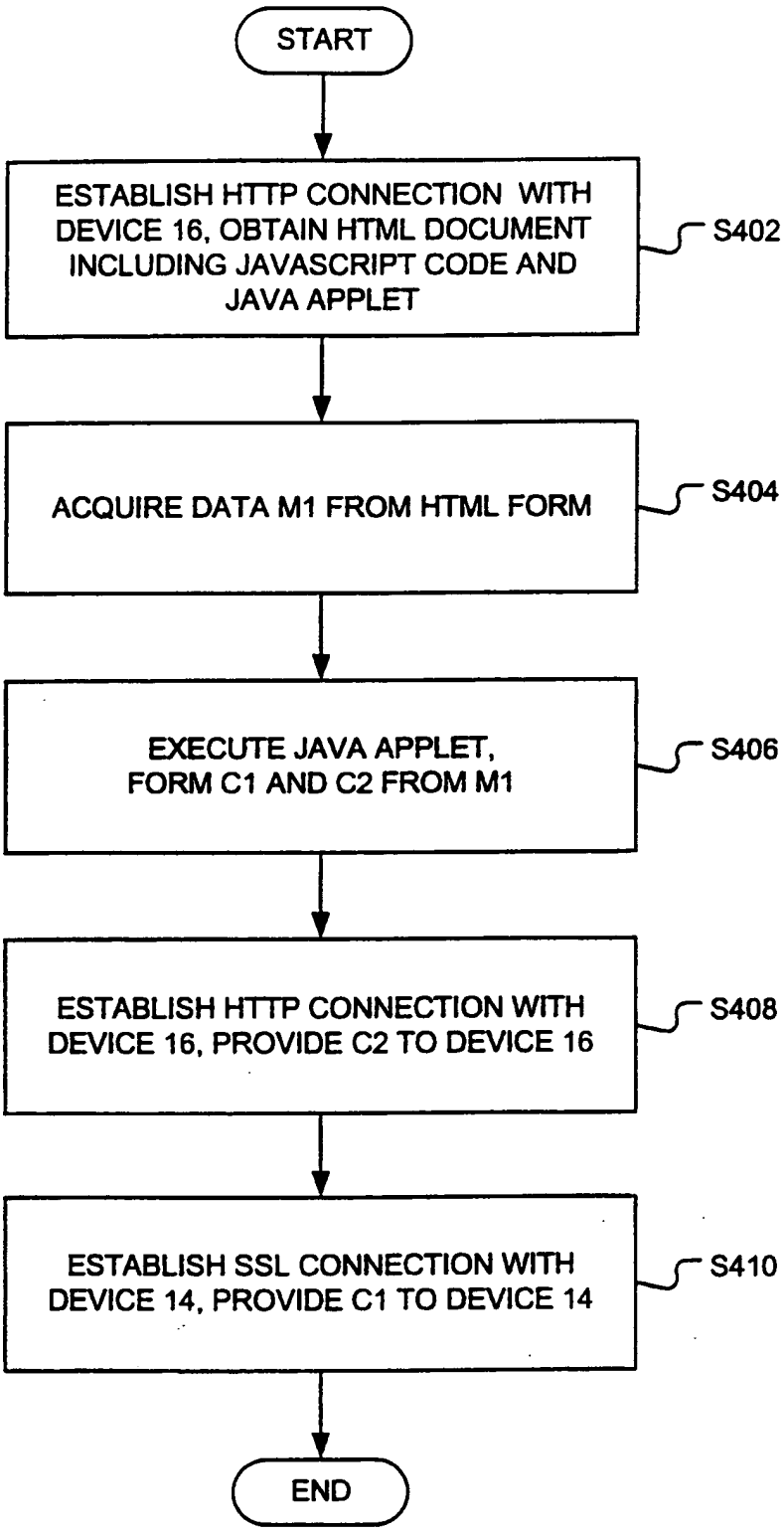

START
400
ESTABLISH HTTP CONNECTION WITH DEVICE 16, OBTAIN HTML DOCUMENT INCLUDING JAVASCRIPT CODE AND JAVA APPLET
S402
ACQUIRE DATA M1 FROM HTML FORM
S404
EXECUTE JAVA APPLET, FORM C1 AND C2 FROM M1
S406
ESTABLISH HTTP CONNECTION WITH DEVICE 16, PROVIDE C2 TO DEVICE 16
S408
ESTABLISH SSL CONNECTION WITH DEVICE 14, PROVIDE C1 TO DEVICE 14
S410
END

FIG. 4

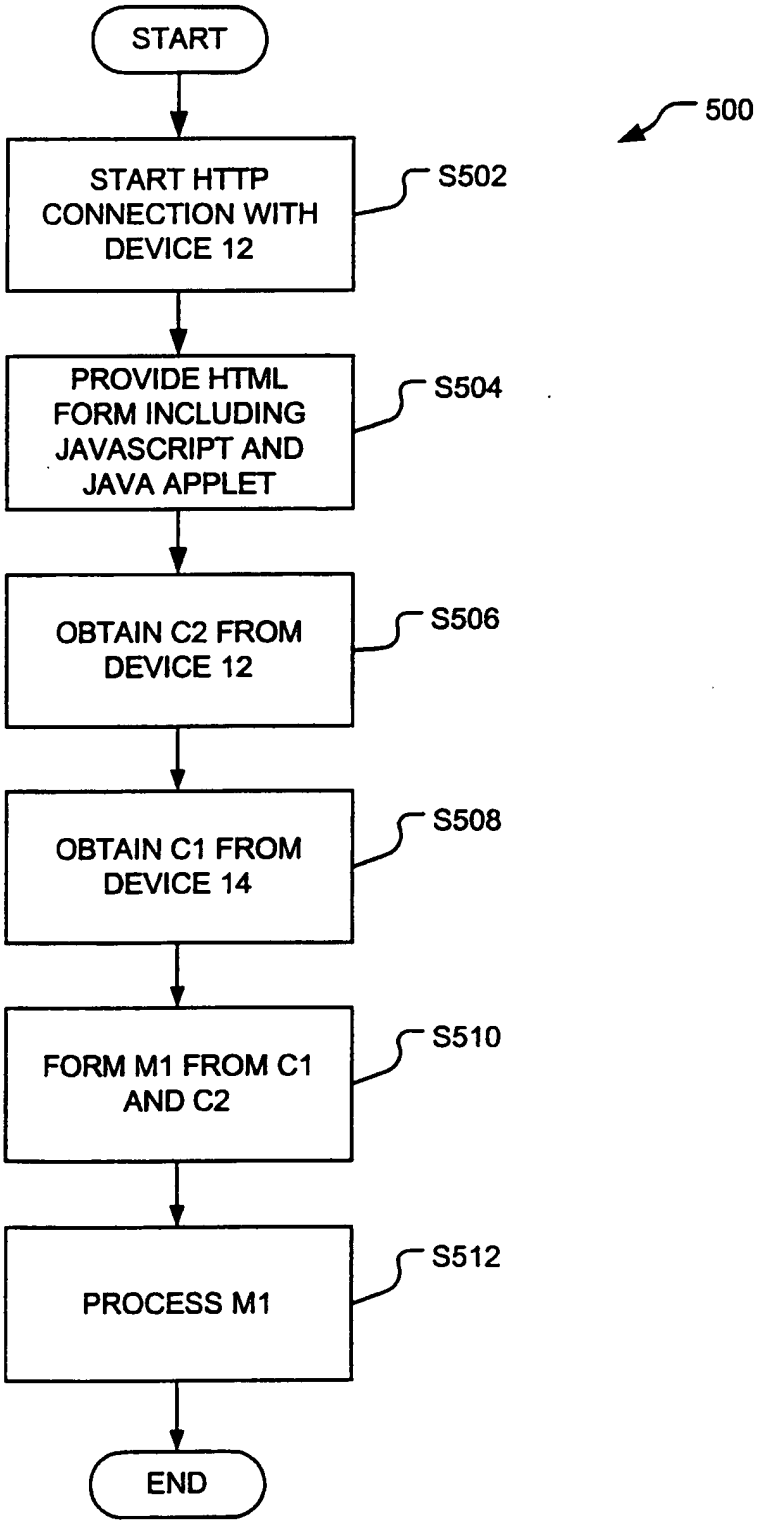

START
500
START HTTP CONNECTION WITH DEVICE 12
S502
PROVIDE HTML FORM INCLUDING JAVASCRIPT AND JAVA APPLET
S504
OBTAIN C2 FROM DEVICE 12
S506
OBTAIN C1 FROM DEVICE 14
S508
FORM M1 FROM C1 AND C2
S510
PROCESS M1
S512
END

FIG. 5

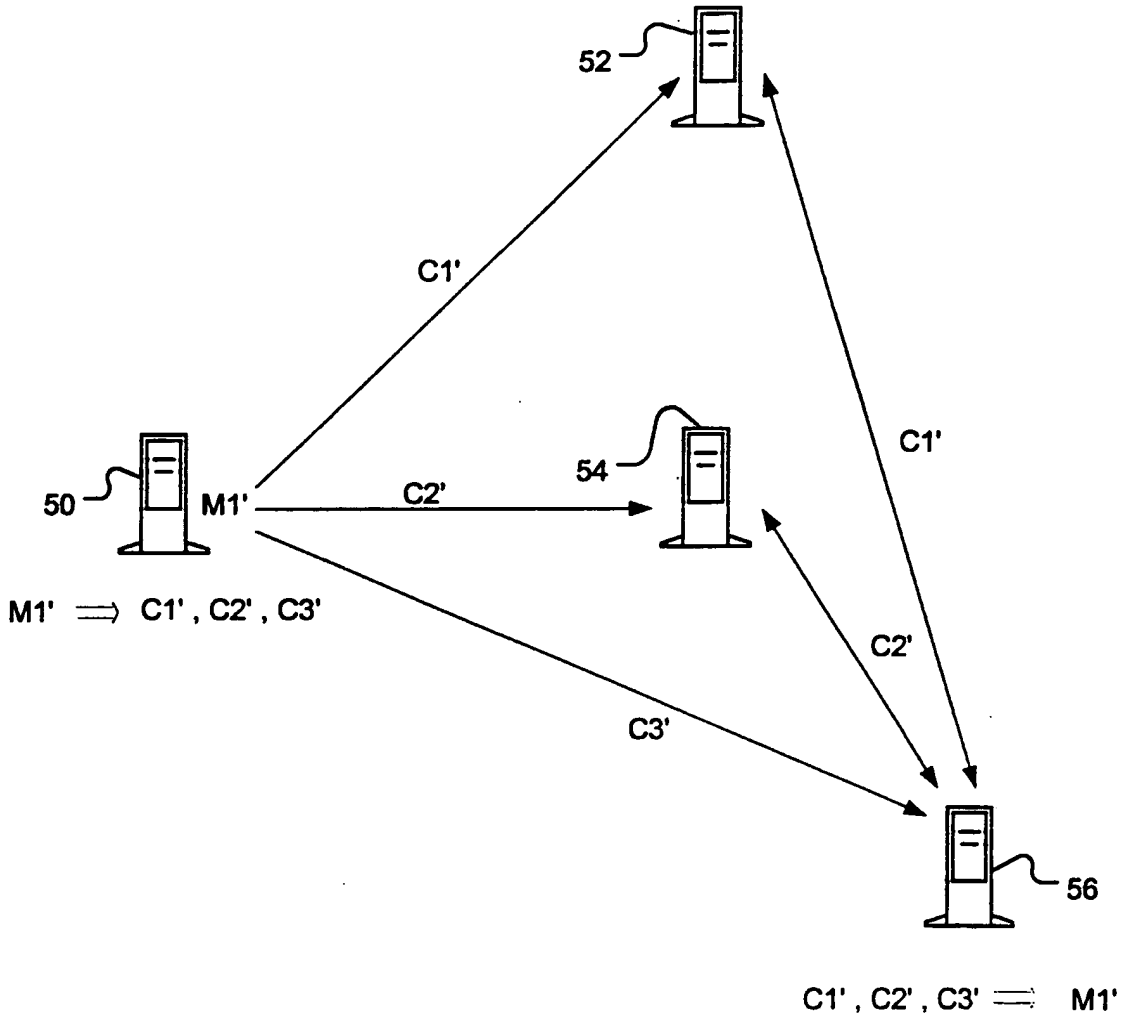


FIG. 6

INTERNATIONAL SEARCH REPORT

Inter onal Application No
PCT/CA 99/00838

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04L29/06

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
P,X	GB 2 332 833 A (INTERACTIVE MAGAZINES LIMITED) 30 June 1999 (1999-06-30) abstract page 1, line 1 - line 3 page 1, line 25 -page 2, line 14 claims 1-12	1-19
A	WO 96 29667 A (SANDBERG DIMENT ERIK) 26 September 1996 (1996-09-26) abstract page 2, line 1 - line 23 page 4, line 12 - line 24; figure 2	1-19

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
24 January 2000	02/02/2000

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Adkhis, F

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

..formation on patent family members

Inter ..nal Application No
PCT/CA 99/00838

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
GB 2332833 A	30-06-1999	AU 1775099 A	19-07-1999
		WO 9934547 A	08-07-1999
WO 9629667 A	26-09-1996	US 5826245 A	20-10-1998
		AU 5366096 A	08-10-1996